IAC-11-D1.6.8

SysML BASED SYSTEM ENGINEERING: A CASE STUDY FOR SPACE ROBOTICS SYSTEMS

S. Chhaniyara, C. Saaj
Surrey Space Centre, University of Surrey, UK, s.chhaniyara@surrey.ac.uk, c.saaj@surrey.ac.uk
B. Maediger, M. Althoff-Kotzias, B. Langpap, I. Ahrns
EADS Astrium GmbH, Space Transportation, Germany,
{bernd.maediger, marianne.althoff-kotzias, bernd.langpap, ingo.ahrns}@astrium.eads.net

Space systems are similar to their terrestrial counterparts in many respects but the main distinction that makes space system unique are due to the harsh and low gravity environment that space systems needs to survive, requirements of system redundancy due to lack of on orbit maintenance or parts replacement and costs associated with those. These put extra emphasis on systems and requirement engineering from the very early stage of systems development lifecycle. Typical space missions comprise of many interconnected systems and systems of systems. Each of these systems need to be satisfied or adhered to thousands of requirements.

Traditional System Engineering (SE) approaches require updating and tracking requirements against their functional or behavioural components manually. On top of that, during early design review stages, mission system engineers may also needs to carefully modify or delete requirements without compromising effects of that on other interconnected or sub systems. This is a very time consuming and complex procedure especially when multiple stakeholders and teams of engineers involved locally or globally. This paper introduces the implementation of Systems Modelling Language (SysML) for modelling complex space robotic systems in context of On-orbit Serving (OOS) missions. In this paper, the benefits of applying Object Management Group (OMG) System Modeling language (SysML™) to support the specification, analysis, design and verification to space robotic systems is being proposed.

I. INTRODUCTION

Robotic technologies are finding increasing benefits for their applications in the field of planetary explorations, autonomous free flying spacecraft, on-orbit serving of satellites and low earth orbit infrastructures (lifetime expansion or functional upgrades). The success of Space robotic missions relies heavily on the performance of many interconnected systems and systems of systems. However, the introduction of robotic technologies is very complex, as it would involve executing highly non-linear systems semi-autonomously or autonomously. Deep understanding of the mission requirements, accurate system modeling and effective communication between systems, systems of systems and the outside world are critical to the success of these missions. In order to realise these missions, thorough analysis should commence right at the foundation level, i.e. at the 'systems' level.

System engineering (SE) is an interdisciplinary field of engineering. There is no standardised architecture for system engineering processes but over the years many SE standards emerged, that makes up system engineering approach. Software engineering processes also evolved parallel to the SE processes but recent process guidelines and standards emphasise the need for integrating both these processes [1]. As shown in Fig. 1, SE process broadly consists of requirements analysis, functional definition, physical definition and design validation phases in one or another form. A typical space mission comprise of many interconnected systems and systems of systems. Each of these systems needs to satisfy or adhere to hundreds of requirements.

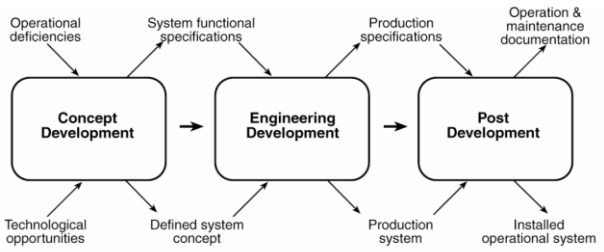

Figure 1 Principle phases of system engineering process life cycle [2]

A SysML based SE process (SysML SEP) for a space robotic system is introduced in [3]. A case study on SysML based modeling of a multimodal sensor system, which is part of the project INVERITAS is presented in [3]. INVERITAS stands for Innovative Technologies for Relative Navigation and Capture of Autonomous Systems. (Project sponsored by the German Space Agency, #50RA0908). This paper extends the work presented in [3] and introduces the automated requirement verification and tracing methodology based on SysML that could drastically reduce the costs, efforts and burden of engineering a space system.

In this paper, Section II provides an introduction to the background work, Section III introduces the SysML-SEP lifecycle, Section IV presents structural and

behavioural SysML model of a close range relative navigation for satellite servicing mission. This section also presents SysML based automated requirement verification and tracing methodology. Finally, a conclusion is presented in Section VI.

II. BACKGROUND WORK

One emerging approach to address the complexity associated with SE of space robotic missions, is to employ a Model Based System Engineering (MBSE), where model is the central artefact. MBSE is a methodology that is the collection of processes, methods and tools to enhance the system engineering process through a model driven approaches [4]. In MBSE, models have a governing role in the requirement engineering, specification, design, integration, validation and operation of a system. MBSE is a paradigm shift from traditional document-based and acquisition lifecycle model approaches. It provides an integrated framework, reduce risk, help to ensure system meet customer requirements, reduce program costs and facilitate trade-off analysis. MBSE also help to promote re-use and integration with legacy systems.

To support MBSE, INCOSE Object-Oriented Systems Engineering Method (OOSEM) is proposed by OMG. It is a model based approach that uses OMG SysML [5]. SysML is a general-purpose modelling language for systems engineering that is a subset of the Unified Modelling Language (UML) with extensions. It is designed to provide simple but powerful constructs for modelling a wide range of systems engineering problems. A SysML model is a set of elements and relations created as a result of an abstraction process. These elements and relations can be visualised through diagrams. It also provides different views of a model and a SysML View is a set of infomation that describes a partial and particular set of system elements and their relationship among them, which is defined by the stakeholders need. It is particularly effective in specifying system requirements, structure, behaviour and allocations and constraints on system properties to support engineering analysis [5]. Moreover, SysML is a powerful language that can capture structural, behavioural aspects and parametric analysis of system under consideration as well as help to test and verify the system.

In [6], an overview of SysML and how this language could help products and systems development through a small case study of rain sensing wiper system is provided. SysML provides an effective mechanism to different aspect of the system and to enforce requirement traceability across it [6]. The growth of in-car embedded electronic systems, driver information systems and development of hybrid and electric cars have increased the system complexity for automobile industry. Extreme pressure to deliver flawless system has driven the use of SysML based design in automobile industry [7], [8]. This new SysML based design methodology has enabled automobile engineers to satisfy automotive safety standards effectively and it is helping engineers to model their system, enhancing the requirements validation and traceability at all level of product lifecycle development [9].

The use of SysML specifically for spacecraft and space system is relatively new. In [10], the author has presented two different approaches to MBSE for phase A and B of spacecraft design and also highlighted several obstacles for implementation of such an environment, like the lack of proper formal semantics and lack of adequate tools for editing the models. It is proposed in [10] that object-oriented modeling languages like UML/SysML can be helpful for spacecraft design. Similarly [11] advocates the use of SysML for satellite system modeling to deal with systems of systems, complex issues, manufacturing demands and to keep risks manageable. In [11], author also echoes that by employing SysML as satellite architecture description language, it enables information reuse between different satellite projects.

Several space agencies are facing challenges to manage growing system complexity and that has led them to invest resources for development of MBSE [12], [13]. As the space agency progress from fly-by to orbital study, on-orbit servicing and in-situ exploration, the system which must accomplish these missions grow in complexity [14]. Further due to ever increasing available processing power has enabled development of high bandwidth sensors, control algorithms. These have resulted in difficulty to communicate complex system architecture and behaviours in textual or static visual forms. Burden on test and verification is tremendous and sometimes it has become so large that it is impractical to fully test systems and system of systems [14]. The result of this situation is that inadequately-specified and incompletely-tested system level interactions are a major and growing risk factor for any mission. Both ESA and NASA are investing and researching SysML based SE processes to cope with future space robotic mission demands [13], [14]. It has been shown that the simulation of complete space system is feasible and can be built by modular model building blocks using SysML [15]. However, in [3] it is shown that there is a need for stringent in house SysML based SE process for complete industry wide applicability.

III. SYSML-SEP

A significant drawback of SysML is that the language is semi-formal and would require defining strict modeling guidelines as described by [3], [16], [17] and [18]. Literature shows that two system engineers independently working on the same system could come

up with totally different SysML models. This shows that there is a need of a robust and stringent in-house best practise and SysML-SEP guidelines. The main aim of [3] is to present a SysML-SEP workflow and develop a modular and reusable SysML user profile to assist in the SE process for a multi-modal sensor system within the German-internal project INVERITAS and for future Astrium Space Transportation (AST) space robotic missions. AST is one of the main users and supporters of static analysis tools in the European space domain[19].

The deployment of the SysML modeling language for capturing system requirements allocated to the software is in progress at AST [19]. During this study, the main focus was concentrated on 3 topics related to software engineering namely, type checking, abstract interpretation and model checking. Further to these efforts in software engineering at AST [16], the results presented in [3] and this paper would elaborate the SysML guidelines and extend it for complete SE for robotic spacecraft. The SysML-SEP follows the process as proposed by [20] and [3]. .A more detailed modular package structure and lifecycle is illustrated in [3].

IV. CASE STUDY – INVERITAS

The long term aim of this research is to investigate the suitability of SE based on SysML standards for modelling multi-modal sensor system (MMS) and controller interface of project INVERITAS using SysML requirement diagrams, use cases, structural and behavioural diagrams. INVERITAS system is the prototypic realization of a broad spectrum of autonomous rendezvous and capture (RvC) systems and the development of the necessary core technologies for improving of the technology readiness level for future OOS missions. The sensor-controller loop is a core building block of all space robotic missions and hence this study will focus on developing generic system architecture for such system level modelling. These generic SysML building blocks then can be reused or adapted for other space applications.

MMS is a system of multi-modality sensory input, data fusion, data exchange and representation. Multi-sensor fusion may result in unified perceptual experiences that are coherent across sensory modalities. The resulting information from multimodal sensor system provides multi layered information which is in some sense better than would be possible when these sources were used individually.

In this research, IBM Rhapsody is chosen as a tool for modelling INVERITAS system in SysML due to its ability to perform integrated requirements traceability and real-time validation of system model [3]. In this paper, top level SysML model of close-range approach servicer satellite to the client satellite is presented and visual navigation sub system is modelled. Visual navigation system consists of Monocular cameras, Stereo-cameras, LIDAR, Image processing unit and illumination device.

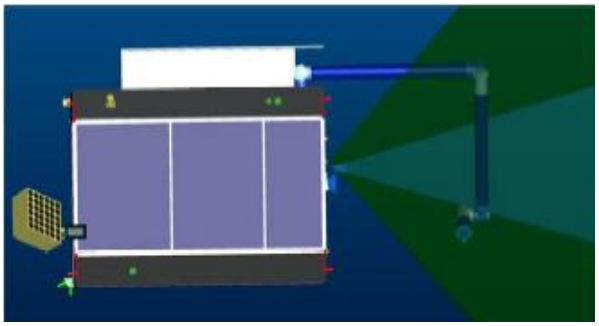

Figure 2 The servicer satellite with field of view of sensors looking at the forearm of the manipulator [21]

IV.I Architectural Model

A context diagram is a top level block diagram showing the architecture of the entire system. It shows the visual navigation module as a part of an Instrument Control Unit (ICU). The visual navigation block and its relationship with the other modules in the system have been shown Figure 3. The visual navigation block is connected to the Rendezvous mission and equipment manager (RMEM), Measurement pre-processing (MP), Guidance Navigation and Control (GNC), Relative measurement (Rel. Mes.), Visual inspection and Abs. Mes. units directly as shown in Figure 3. Whereas, Flight control monitoring (FCM) assists the visual navigation block indirectly through GNC and MP.

IV.II Internal Block Definition Diagram (IBD)

An IBD shows the internal architecture of a system. It uses parts and connectors to represent the architecture. IBD helps to identify different interfaces between various elements of a system. The IBD of ICU is shown in Figure 4 and IBD of Camera control unit (CCE) shows preliminary imager processing functions and different interfaces between those.

IV.II Use case diagram

Use case diagrams define the various scenarios that a system can have. The use case communicates the basic idea of how the system would act under different conditions and which stakeholders are responsible for a particular use case. Use case analysis forms basis of system design and system behaviour. The top level use case for Visual navigation MMS is presented in Figure 8. For each use case it is important to identify prime actor, trigger, preconditions, nominal scenario scenarios and any contingency scenarios if exist. Use case analysis helps to develop behaviour of a system which could be developed in the form of sequence diagram, state-machine diagram or activity diagrams.

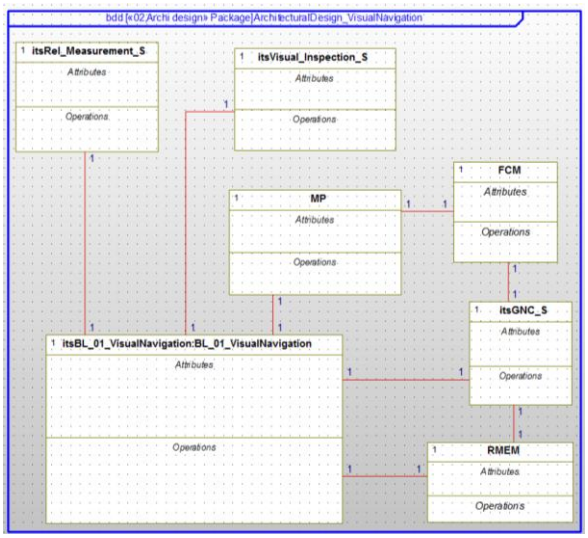

Figure 3 Context block diagram of INVERITAS_ VisualFunctions at the First level

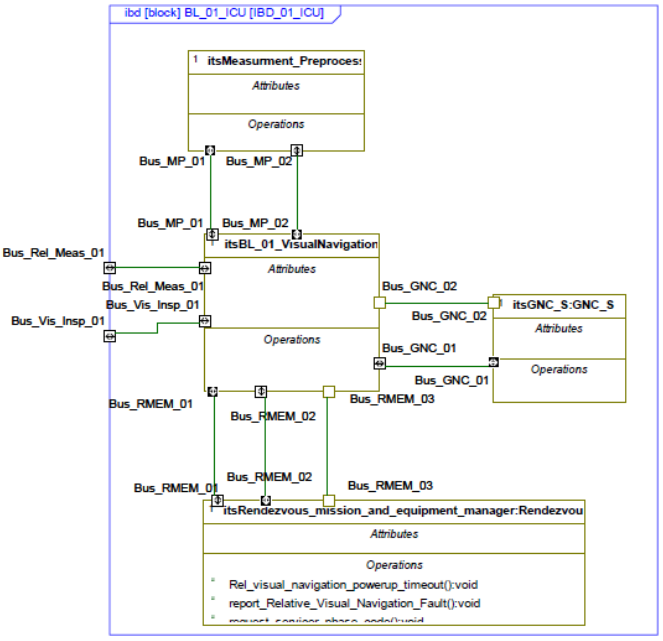

Figure 4 IBD of ICU and interfaces between its elements

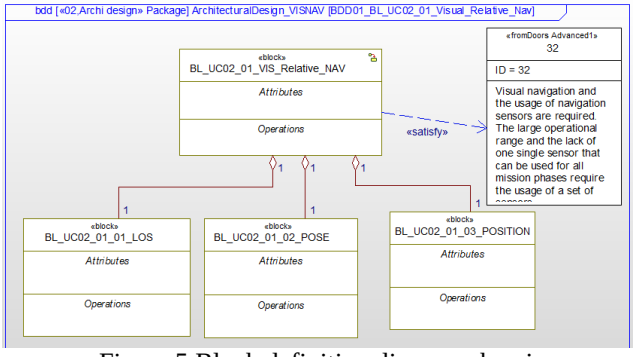

Figure 5 Block definition diagram showing BL_UCO2_Vis_Relative_Nav

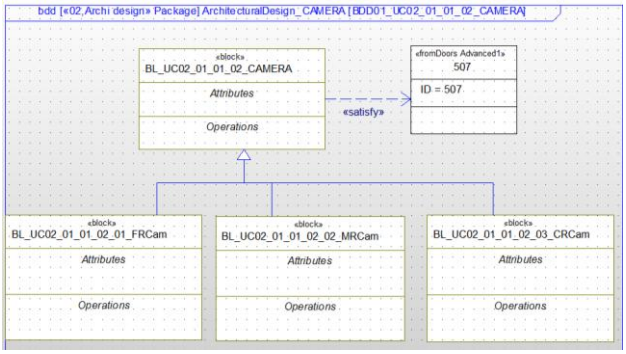

Figure 6 Block definition diagram of BL_UC02_01_01_02_CAMERA

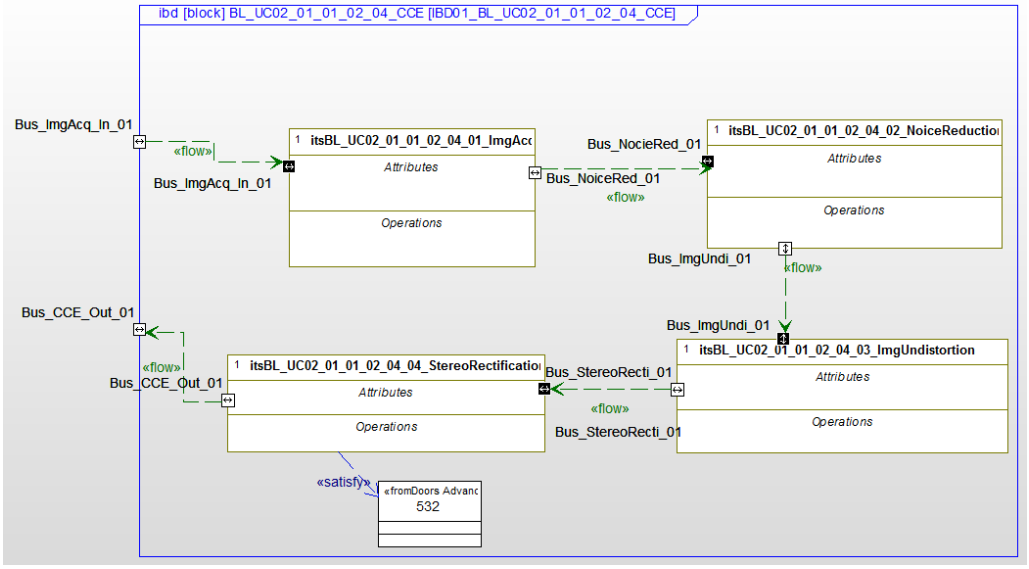

Figure 7 IBD of CCE for the Close range phase scenario

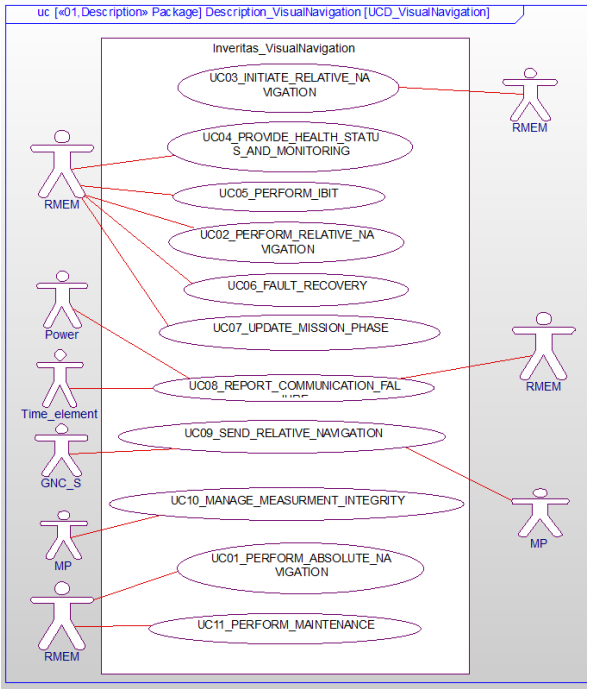

Figure 8 First level use case: UCD_VisualNavigation

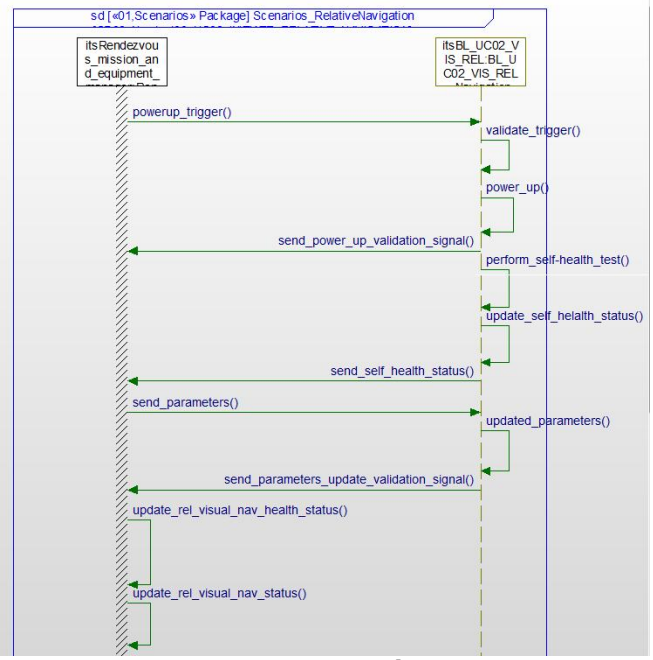

Figure 9 SD03_Nominal00_UC03_ INITIATE_RELATIVE_NAVIGATION

IV.III State Machine diagram

SysML state machine diagrams describe the behaviour of a block over time through illustrations of the states and transitions of a single instance of a block progressing through its lifetime. State machine diagrams are a traditional object-oriented way to show behaviour and to document how an object responds to events, including internal and external stimuli. The first iteration of a state diagram for VIS_REL_Navigation block is shown in the and the different states and corresponding events that causes state changes are shown.

IV.IV Sequence diagram

A sequence diagram shows the specific interactions in terms of control flow, defined by sending and receiving message (control and data) between collaborating elements of a system. The timing ordering of the message is indicated by vertical placement of the message on lifeline of each object in the diagram. Use cases are a description of high-level functionality and use case analysis could be used to identify different scenarios of a sequence diagram both nominal and contingency. A sample sequence diagram showing initialisation of relative navigation between RMEM and VisualNavigation block is shown in Figure 9.

Similarly, detailed architectural and behavioural aspect of this Visual Navigation module of INVERITAS is created as per customer requirements and relevant modeling elements are linked with requirements through satisfy/trace relationships. The detailed requirement verification and tracing procedure is presented in the section V.

IV. REQUIREMENT TRACING

Modeling requirements with SysML helps managing system complexity from early design stage. Requirements can be decomposed into atomic requirements and may later even be related in the sense that together they are capable of delivering a whole feature. Grouping of requirements can be achieved through SysML package/profile management as described in [3].

Requirement traceability is defined as: “the ability to describe and follow the life of a requirement, in both a forward and backward direction, i.e., from it’s origins, through its development and specification to its subsequent deployment and use, and through periods of on-going refinement and iteration in any of these phases” [22]. System engineers typically need to show traceability of requirements between customer documents and specifications. It is essential that they be able to identify what portions of their systems design satisfies specific requirements. Moreover, understanding the ramifications of adding or deleting requirements in complex designs, or ascertaining which requirements drove the creation of certain design elements, can be a daunting task.

Customers typically provides higher level requirements either in word, excel or IBM Dynamic Object Oriented Requirements System (DOORS) format. Particularly, Defence and Space domains use of DOORS for requirement management and traceability

analysis is widely prevalent. The requirements management task is performed within Rational DOORS.

Rhapsody for Systems in conjunction with Rhapsody Gateway provides a world class, and extremely flexible approach to managing, linking and tracing requirements from the repository to inside the model. The Rhapsody Gateway imparts the benefits of a seamless bi-directional information exchange interface with third party requirements and authoring tools to extend a complete traceability solution, which allows developers to examine the upstream and downstream impact of requirements changes, in real time, at any level [23]. The tool is flexible, offering an open architecture structure that supports multiple workflows and is engineered to organize complex requirements scenarios. In particular, Gateway helps companies meet the demanding defence and space safety standard, to name but one industry standard the tool supports. DOORS has been used as a prime tool for this INVERITAS project. It is suggested to use Rhapsody Gateway add on to link DOORS with the Rhapsody SysML mode and to accelerate requirements tracing and verifications.

The requirements management task is performed within Rational DOORS. Typically, Rational DOORS maintains project documents, user documents, and documentation of changes. System specification and modeling are performed within Rational Rhapsody. The model is built, however, to meet the requirements stored in Rational DOORS. Prototyping and analysis done in Rational Rhapsody verify that the model is consistent with the provided requirements. The Gateway interface works by sharing information between the Rational Rhapsody model and the Rational DOORS database [23]. The major steps for requirement tracing using Rhapsody Gateway is presented as per below:

Step 1: Rhapsody gateway project configuration

Rhapsody Gateway launches the Configuration dialog box which provides a selection menu item or toolbar button to configure one of the following parts of a project: Project, Types, Snapshots, Filters, Reports, Expressions, XML or Options. This configuration window helps to customise links between SysML model, DOORS requirement database, documents and also gives full control of types of information that each document can bring to the Gateway project. This configuration setup also could be used to control the synchronisation and tracing of specific requirements through control attributes defined in DOORS.

Step: 2 On applying the configuration settings, Gateway project synchronised. Then higher level requirements to the Rhapsody SysML model can be added to upward_requirements folder of SysML model [3].

Step: 3 Perform requirements allocations task where relevant requirements can be attached to corresponding model element/elements through dependency relationships in the Rhapsody SysML model.

Step: 4 Go to Rhapsody Gateway's Management view and check coverage statistics

The Gateway management View contains project information. The Overall Quality area within management view displays the analysis results according to requirements. A status bar shows the ratios of errors and warnings in relation to the total requirements number. Depending on the analysis required, Gateway add on provides Coverage analysis view, Impact analysis view, graphical view, requirement details and link details view [23]. These views are briefly explained here below for the INVERITAS_VisualFunctions project.

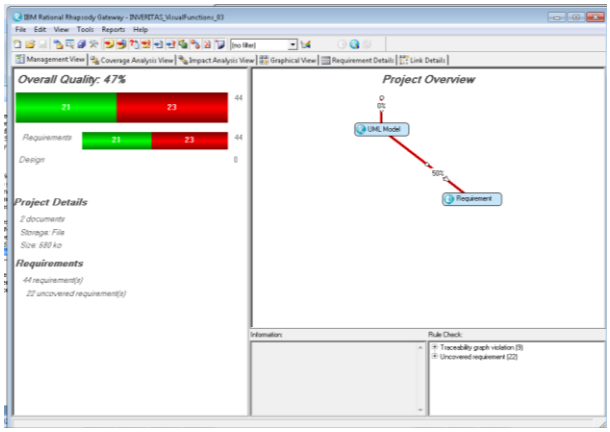

Figure 10 Management view - Rhapsody gateway

V.I Coverage analysis view

The Coverage Analysis view displays for a selected element of a document and displays requirement coverage one level upstream and one level downstream from the selected document as shown in Figure 10.

- Upstream Coverage Information- Displays one level of covered requirements, N-1, where N is a selected document element in the selection column.
- Downstream Coverage Information- Displays one level of covering requirement reference elements, N+1, where N is a selected document element in the Selection column.

V.II Impact analysis view

The Impact Analysis view displays for a selected element of a document, for all levels of covering elements, N-m, and for all levels of covered elements, N+p, from other documents as defined by the project.

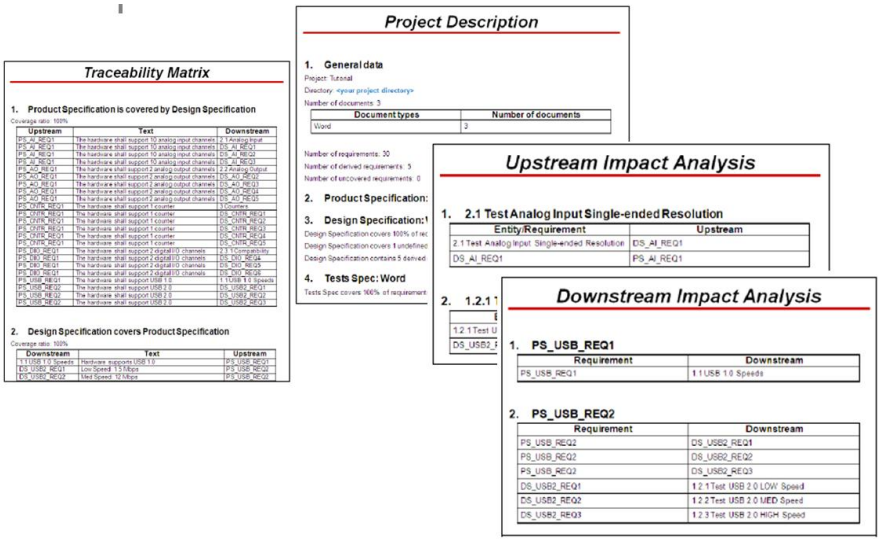

Figure 11 Automated requirement tracing report generation

- Upstream Impact Information—Displays all levels of covered requirements, N-m, for a selected document element in the Selection column.
- Downstream Impact Information—displays all levels of covering requirement reference elements, N+p, for a selected document element in the Selection column.

V.III Graphical view

The Graphical View displays each document as an object with its traceability elements displayed in a tree view within the object. The Graphical view can display the full traceability graph also when a particular element being selected, the view highlights the selected element, the covering elements and the lines between the elements, as shown in the. The graphical view can also display a partial graph by focusing one some documents elements or by hiding some documents.

V.IV Requirement details view

The Requirement Details view displays each requirement and its attributes for a document in a table. The document displayed is selected from the drop-down list box in the upper left. For this document, all the requirements, derived requirements, macro-requirements and entities are presented in the first column. The other column headers are the attributes and the cells contain values of these attributes.

V.V Links details view

The Link Details pane displays covering information between a covering document and its covered documents.

V.VI Summary of Requirement traceability

When the Rational Rhapsody Gateway analyses project information including requirements, documents, and database modules, the software provides the following analysis results [23]:

- Traceability of requirements workflow on all levels, in real time
- Automatic management of complex requirements scenarios for intuitive and understandable views of upstream and downstream impacts
- Creates impact reports and requirements traceability matrices to meet industry safety standards
- Connects to common requirements management and authoring tools including IBM Rational DOORS®, IBM Rational Requisite®, Microsoft® Word, Microsoft Excel, ASCII, Adobe® FrameMaker, Code, and Test files
- A bidirectional interface with the third-party requirements management and authoring tools
- Monitoring of all levels of the workflow, for better project management and efficiency
- Rhapsody Gateway allows compliance with the traceability objectives defined by quality standards including ECSS-E40 for space.

VI. CONCLUSION

This paper highlights the need of MBSE process for complex space robotic projects and presents a case study on SysML based SE process for MMS system of project INVERITAS. SysML has an important drawback that the language is semi-formal and would

require defining strict modeling guidelines to ensure streamlined model creation and organisation of complex projects. This paper contributes towards demonstrating automated requirement tracing and verifications.

This study provides further insight into SysML standards and its applicability for complex space projects in industrial environment. Future work would involve further detailed second and third level functions behaviour analysis and simulation of close-range phase of OOS mission and to demonstrate how SysML could help with simulation, test case generations and trade-off analysis.

II. REFERENCES

1. Boehm, B. Some Future Trends And Implications For Systems And Software Engineering Processes. Systems Engineering, V. 9, N. 1, P. 1-19, 2006.
2. Kossiakoff, A.; Sweet, W. N. Systems Engineering: Principles And Practice. [S.L.]: Wiley-Interscience, 2003.
3. Chhaniyara, S. Et Al. Model Based System Engineering For Space Robotics Systems. Esa/Estec. [S.L.]: Esa. 2011.
4. Fisher, J.; Others. Model-Based Systems Engineering: A New Paradigm. Incose Insight, P. 3-16, 1998.
5. Sysml, O. Systems Modeling Language (Sysml) Specification Final Report. Object Management Group, 2007.
6. Balmelli, L.; Others. An Overview Of The Systems Modeling Language For Products And Systems Development. Journal Of Object Technology, V. 6, N. 6, P. 149-177, 2007.
7. Boulanger, J. L. Experiences From A Model-Based Methodology For Embedded Electronic Software In Automobile. Ieee. [S.L.]: [S.N.]. 2008. P. 1-6.
8. Guo, Y.; Rao, A. C.; Jones, R. P. Architectural And Functional Modelling Of An Automotive Driver Information System Using Sysml. Ieee. [S.L.]: [S.N.]. 2008. P. 552-557.
9. Rao, A. C. Et Al. Systems Modelling Of A Driver Information System-Automotive Industry Case Study. Ieee. [S.L.]: [S.N.]. 2006. P. 6--Pp.
10. Hein, A. M. Et Al. Object-Oriented System Models In Spacecraft Design: First Steps Towards An Application In Phase B. [S.L.]: [S.N.]. 2010.
11. Leonor, B. B. F.; Santos, W. A. D.; Stephany, S. A Knowledge-Based Approach To Deal With Conceptual Satellite Design. [S.L.]: Springer-Verlag Berlin, Heidelberg 2009. 2009.
12. Dvorak, D. Nasa Study On Flight Software Complexity Final Report. Nasa Office. [S.L.]. 2009.
13. Krueger, T. Modelling Of A Complex System Using Sysml In A Model Based Design Approach. Esa/Estec. [S.L.]: Esa. 2011.
14. Bayer, T. J. Et Al. An Operations Concept For Integrated Model-Centric Engineering At Jpl. Ieee. [S.L.]: [S.N.]. 2010. P. 1-14.
15. Raif, M.; Walter, U.; Bouwmeester, J. Dynamic System Simulation Of Small Satellite Projects. Acta Astronautica, V. 67, N. 9-10, P. 1138-1156, 2010.
16. Hiron, E.; Miramont, P. Process Based On Sysml For New Launchers Sytem And Software Developments. Data System In Aerospace: Dasia 2010. Budapest, Hungary: [S.N.]. 2010.
17. Ober Iulian, D. I. Unambiguous Uml Compsite Structures: The Omega2 Experience. International Conference On Current Trends In Theory And Practice Of Computer Science. Lncs: Springer. 2011. P. 418-430.
18. Ober Iulian, D. I. A New Version Of The Profile And The Tools. Uml & Aadl- 15th Ieee Iceccs. [S.L.]: Ieee. 2010. P. 373-378.
19. Lesens., D. Step Software Engineering – Progress Report. Eads Astrium Space Transportation, Les Mureaux. [S.L.]. 2010.
20. Hoffmann, H. P.; Telelogic, A. I. B. M. Sysml-Based Systems Engineering Using A Model-Driven Development Approach. [S.L.]: [S.N.]. 2006.
21. Team, R. Deos Phase B, Rendezvous \& Deorbiting, Phase B Design Definition. Astrium St, Astrium-Sl, Dlr-Rb, Ifr, Jop, Vh\&S. [S.L.]. 2010.
22. Gotel, O. C. Z.; Finklestein, A. C. W. An Analysis Of The Requirements Traceability Problem. Icre94: 1st International Conference On Requirements. Colorado Springs: Ieee Cs Press. 1994.
23. Ibm. User Manual - Ibm Rational Rhapsody Gatewa Add On. Ibm. [S.L.]. 2010-2011.